THESIS

PRECONDITIONING POLYNOMIAL SYSTEMS FOR HOMOTOPY CONTINUATION

Submitted by

Steven L Ihde

Department of Mathematics

In partial fulfillment of the requirements

For the degree of Master of Science

Colorado State University

Fort Collins, Colorado

Summer 2011

Master's Comittee:

Advisor: Dan Bates

Chris Peterson
Peter Young

ABSTRACT

PRECONDITIONING POLYNOMIAL SYSTEMS FOR HOMOTOPY CONTINUATION

Polynomial systems are ubiquitous in today's scientific world. These systems need to be solved quickly and efficiently. One key solution method comes from Numerical Algebraic Geometry, specifically Homotopy Continuation. This method involves following paths from the solutions of a simpler system to the solutions of the target system. If we can follow fewer or better conditioned paths to the solution set, the result is better efficiency. Our goal is to precondition the original system in order to achieve such efficiency. Using dual spaces and H-bases, we are able to remove poorly conditioned paths and at worst replace them with, possibly more, better conditioned paths. At best we can trim the system down so that we track only the paths that lead to solutions. These techniques require only numerical linear algebra and are therefore easily computed. In this thesis we will introduce H-bases and dual spaces, show some promising preliminary results, and discuss further work in this area.

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

The idea of a system of polynomial equations can be found throughout scientific literature. These systems of equations arise everywhere from systems biology and chemical reaction networks to robotics and engineering. The efficient solution of these equations can help push the science further. There are several different methods throughout mathematical literature that work at solving such systems of equations. One area is Numerical Algebraic Geometry, specifically the area of algebraic geometry involving numerical methods. One of the core ideas in this area is that of homotopy continuation. We will briefly define and discuss homotopy continuation in the next chapter but the main idea is simple. We start with a system of equations that we want to solve. We solve a simpler system and build a homotopy, or continuous morphing, from the simpler system to the target system. We must be careful, though, as this can cause problems when the paths that we follow are ill-conditioned.

1.1 Root Counts

One question that has driven much research throughout the history of algebraic geometry is that of root counts. That is, we are interested in the number of zeros, or roots, of a polynomial. By the Fundamental Theorem of Algebra, we have that a polynomial of degree d in one variable over $\mathbb{C}$ must have exactly d roots [5].

Once we move beyond a single variable, the question becomes far more interesting. The first way to count comes from a theorem of Bézout [3]. This is the *total degree* count, an upper bound on the number of distinct, isolated solutions to a system of polynomials is determined by the product of the degrees of the polynomials. So, a system of 6 equations each of degree d would have a total degree count of d^6. While this is a most efficient way to compute an upper bound, it is rarely sharp.

Another type of count is the *mixed volume* calculation. This count uses Newton polytopes built by the degrees of all of the monomials of each polynomial. Each polynomial in a system creates its own associated Newton polytope. At this point a calculation called a Minkowski sum "adds" the polytopes in order to create one large polytope. The volume of this new polytope, called the mixed volume, is another upper bound for the number of roots [3]. This count is more effective and is often much lower than that of the Bézout bound. For sparsely populated systems, this method is very efficient. However, if the system is too large, or there are too many monomial terms, the mixed volume calculations become unwieldy. Even for moderately sized problems, the calculation can easily become unmanageable. Part of the problem is that these root counts are generic. They are not specific to each system. As a result, while the total degree is easy to compute and the mixed volume is often a very good bound, they are still just upper bounds.

1.2 The Idea

Associated to each of these root counts is a specific homotopy. Due to these generic root counts, one problem that arises using homotopy continuation is that the simple start system may create many more paths than there are solutions. For example, consider a start system based on a total degree (or Bézout) count

for the number of roots. Since this is generally a large upper bound, some of the extraneous paths may lead off to ∞. These infinite paths are computationally expensive to track as they must be followed to a predetermined tolerance in order to determine that they are infinite. Then, they must be truncated at some point which may lead to the loss of a solution if the bound was not correctly set. Our goal is to precondition the original system to cut down the number of paths being tracked. We are especially interested in removing the paths that go off to ∞. If we can inexpensively precondition the system to remove paths at ∞, we can save computational cost. This allows for the solving of larger systems in a reasonable amount of time and with less computing power.

We are going to use dual spaces and H-bases to precondition these systems. By homogenizing our system and moving to the dual space of the system, we are able to cut away these infinite paths. As an added bonus, if we follow this enough steps, we are able to read off the exact number of solutions. We will be saturating the ideal generated from our system by the homogenizing variable in order to remove extraneous points. These computations, normally computed using Gröbner bases, will be reduced to linear algebra and numerical linear algebra computations. The Gröbner calculations, usually computed in a computer algebra system such as *Macaulay 2* [6], can break down when a system becomes too large since they tend to grow combinatorially. The fact that these dual space computations are essentially linear will allow for the solution of larger problems with less computational cost. Another benefit of the methods using only numerical linear algebra is that they are naturally parallelizable. This is joint work with Dan Bates and Jon Hauenstein.

Chapter 2

GENERAL BACKGROUND

Before we can begin to precondition polynomial systems for solving, we need some background. We want to use ideas from abstract algebra and algebraic geometry to solve these systems of equations. Here we will discuss some preliminary ideas like polynomial systems, polynomial rings, ideals and varieties, and homotopy continuation. The last will be our motivation for preconditioning. We will discuss difficulties that arise using homotopy continuation and how we hope to remove them.

2.1 Ideas from Algebra

We start with some notation. We will be working over the field of complex numbers $\mathbb{C}$. This field is convenient because it is infinite and algebraically closed. It contains the real numbers $\mathbb{R}$ as a subset and allows us to find the solutions we are looking for. From algebra we have that this is a ring since every field is a ring with special properties, specifically it is commutative with identity and has no zero divisors. The usual notation for a general field is k but for our purposes we will always work over $\mathbb{C}$. For more information on rings and fields see [5]. We now want to consider a polynomial ring.

Definition 2.1.1. *A **polynomial ring** R is a ring adjoined by a set of variables* $x_1, \ldots, x_n$,

$$R = \mathbb{C}[x_1, \ldots, x_n].$$

This new ring R is the set of all polynomials of finite degree with coefficients in $\mathbb{C}$. We are assuming here that the reader has a working knowledge of polynomials from elementary algebra.

Definition 2.1.2. *A **polynomial system** is a set of polynomials* $\{f_1, \ldots, f_s\}$ *such that for* $i = 1 \ldots s$ *we have* $f_i \in \mathbb{C}[x_1, \ldots, x_n]$.

We want to put this idea into the context of abstract algebra. Namely, we want to use the *ideal* generated by our polynomial system.

Definition 2.1.3. *We say that a subset* $I \subset R = \mathbb{C}[x_1, \ldots, x_n]$ *is an **ideal** of* R *if*

1. $I \neq \emptyset$
2. I *is closed under subraction and multiplication*
3. $rI = \{ra \mid a \in I\} \subseteq I$ *for all* $r \in R$

For brevity we only use the definition under left multiplication of ring elements. Since we are working with a field for our base ring, we have a commutative ring and the definition is equivalent. Now we want to think about how we can generate an ideal. By the Hilbert Basis Theorem, any ideal of a polynomial ring over a field is finitely generated [5]. We can thus create an ideal I of our polynomial ring R by considering the ideal generated by our system of polynomials. From [4] we know that this will satisfy our criteria for an ideal.

Definition 2.1.4. *Consider a system of polynomials $f_1, \ldots, f_s$ in $\mathbb{C}[x_1, \ldots, x_n]$. The **ideal generated by** $f_1, \ldots, f_s$ is the set of all polynomial combinations of $f_1, \ldots, f_s$ with elements of $\mathbb{C}[x_1, \ldots, x_n]$. Namely,*

$$< f_1, \ldots, f_s > = \{\textstyle\sum_{i=1}^{s} g_i f_i \ : \ g_1, \ldots, g_s \in \mathbb{C}[x_1, \ldots, x_n]\}.$$

With this definition, we have an ideal formed by our system. We want to find a way to "divide" two ideals. This idea will accordingly be called a quotient ideal or, from the notation, a colon ideal.

Definition 2.1.5. *Consider I and J ideals of $R = \mathbb{C}[x_1, \ldots, x_n]$.*
*Then the **quotient ideal** $I : J$ is the set*

$$\{f \in \mathbb{C}[x_1, \ldots, x_n] : fg \in I \text{ for all } g \in J\}$$

We can show that this is in fact an ideal [4]. This idea can be extended to include a single polynomial. If I is an ideal in our ring R and f is a polynomial in R, we can take the ideal quotient $I : < f >$. We make the notation simple by dropping the $<>$ notation, as in $I : f$. The question that remains is, how much can we quotient out using these ideals? Is there a point where we can't divide out any more? We can consider the ideal $(I : f) : f$. If we iterate this process, will it terminate? A proposition from [4] allows us a first step to show this is true.

Proposition 2.1.1. *If I and J are ideals in $\mathbb{C}[x_1, \ldots, x_n]$ then $I \subset I : J$*

Proof:

Since I is an ideal, then for all $f \in I$, $fg \in I$ for all $g \in \mathbb{C}[x_1, \ldots, x_n]$. Then certainly $fg \in I$ for all $g \in J$. Thus $I \subset I : J$

□

Since we have shown that $I \subset I : f$ we have by the same argument,

$$I : f \subset (I : f) : f$$

For notation sake, we will refer to $(I : f) : f$ as $I : f^2$. Now we can build an ascending chain of quotient ideals

$$I \subset I : f \subset I : f^2 \subset I : f^3 \subset \ldots$$

By the ascending chain condition [5, 4], we have that this must terminate. Specifically this means that there exists some ℓ such that for all $m \geq \ell$ we have $I : f^m = I : f^{m+1}$. This leads us to our next and crucial definition.

Definition 2.1.6. *Consider an ideal $I \subset \mathbb{C}[x_1, \ldots, x_n]$ and $f \in \mathbb{C}[x_1, \ldots, x_n]$. Then the **saturation** of I with respect to f, denoted $I : f^\infty$, is*

$$I : f^\infty = \{h \in \mathbb{C}[x_1, \ldots, x_n] \text{ such that } f^m h \in I \text{ for some } m > 0\}$$

This definition will allow us to find a stopping criterion for our algorithm. It is an easy exercise to show that $I : f^\infty$ is an ideal and since there must exist ℓ such that $I : f^\ell = I : f^{\ell+1} = I : f^{\ell+2} = \ldots$ we have that $I : f^\ell = I : f^\infty$.

Now that we are able to put our system into the language of abstract algebra, our next goal is to relate this to the inherent geometry of the system. Since we want to find the solution to our polynomial system, this means that we want to solve all of the polynomials simultaneously. This is equivalent to finding the set of values at which all of the polynomials evaluate to zero. We will call upon a definition from geometry to describe this set of values. Here, again, we will follow the notation of [4].

Definition 2.1.7. *The* ***variety*** *of an ideal* $I = < f_1, \ldots, f_s > \subset R$ *is the set of points* $(a_1, \ldots, a_n) \in \mathbb{C}^n$ *such that* $f_i(a_1, \ldots, a_n) = 0$ *for* $i = 1, \ldots, s$

$$V(I) = V(f_1, \ldots, f_s) = \{(a_1, \ldots, a_n) \in \mathbb{C}^n \mid f_i(a_1, \ldots, a_n) = 0 \text{ for } i = 1, \ldots, s\}$$

This language of algebraic geometry allows us to redefine our solution set. We can now say, for a system of equations $F = \{f_1, \ldots, f_s\}$, the set of zeros is the variety $V(F)$. This of course corresponds to the solution set. For the purposes of this thesis we will only consider varieties whose dimension is 0. This means we will only consider polynomial systems whose solution set is a finite set of points in $\mathbb{C}^n$. For varieties of positive dimension, there is still more work to be done.

2.2 Homogeneous Systems

In this section we consider the algebraic and geometric constructs of the last section with a new twist. We will introduce the idea of working with *homogeneous* systems.

Definition 2.2.1. *A polynomial* $f \in \mathbb{C}[x_1, \ldots, x_n]$ *is* ***homogeneous*** *if all monomials of* f *have the same degree.*

This definition can be taken one step further. Instead of just having a polynomial that is already homogeneous, we want to take an arbitrary polynomial and make it homogeneous. This will be accomplished by taking the individual monomials and increasing their degrees until they match the degree of the polynomial.

Definition 2.2.2. *Consider a polynomial $f \in \mathbb{C}[x_1, \ldots, x_n]$ of degree d with*

$$f = \sum_{j=1}^{N} c_j \bar{x}^{\alpha_j}$$

*with $\bar{x}$ the set $x_1 \cdot \ldots \cdot x_n$, $\alpha_j = (\alpha_{j,1}, \ldots, \alpha_{j,n})$ the vector of powers of $\bar{x}$ and the coefficients $c_j \in \mathbb{C}$. We can **homogenize** f by multiplying each monomial of f by the appropriate power of x_0, called the **homogenizing variable**.*

$$f^h = \sum_{j=1}^{N} c_j \bar{x}^{\alpha_j} x_0^{d-|\alpha_j|} \text{ with } f^h \in \mathbb{C}[x_0, \ldots, x_n]$$

This new homogeneous polynomial lives in a slightly larger polynomial ring, $\mathbb{C}[x_0, \ldots, x_n]$. With this in hand, we want to build an ideal that is also homogeneous. This begs the question: Can we homogenize an ideal of polynomials? The answer is yes. Unfortunately, this is not as simple as homogenizing each of the generators.

Definition 2.2.3. *Consider an ideal $I \subset \mathbb{C}[x_1, \ldots, x_n]$. The **homogeneous ideal** I^h is the ideal generated by the homogenization of all $f \in I$,*

$$I^h = < f^h \mid f \in I >$$

While we always have that $< f_1^h, \ldots, f_s^h > \subset I^h$ for $I = < f_1, \ldots, f_s >$, we can have strict containment. However, another characterization of a homogeneous ideal allows us to realize a homogeneous basis.

Definition 2.2.4. *Consider $I^h \subset \mathbb{C}[x_0, \ldots, x_n]$, then I^h is a **homogeneous ideal** if*

$$I^h = < g_1, \ldots, g_r > \text{ where the } g_i \text{ are homogeneous polynomials}$$

Informally, our homogenizing variable x_0 plays the role of ∞. Now, if we were to solve the system of equations using homotopy continuation, paths that go off to ∞ correspond to solutions where $x_0 = 0$. If we then use saturation to remove the x_0 parts from our homogeneous system during preconditioning, the undesirable paths will be removed. We can consider the ideal generated by x_0 and use the quotient ideal just as we did in the previous section. For example if I^h is a homogeneous ideal in $\mathbb{C}[x_0, \ldots, x_n]$ then we can consider the ideal $I^h : x_0$. Following the notation from the last section, the saturation of I^h with respect to x_0 will be denoted $I^h : x_0^\infty$.

2.3 Homotopy Continuation

In this section, we consider our main tool for solving polynomial equations. It is good to note that this method is not limited to only solutions of polynomial systems. There may , however, be further complications that arise when using this method to solve other types of equations. Homotopy continuation is well behaved for polynomial systems and this is where our interests lie.

We start by considering a map $f : \mathbb{C}^n \to \mathbb{C}^n$. We can implicitly assume that f is a system of polynomials in the polynomial ring $R = \mathbb{C}[x_1, \ldots, x_n]$. If we then choose some $g : \mathbb{C}^n \to \mathbb{C}^n$ that is easy to solve, we can build a *homotopy* between the two polynomial systems.

Definition 2.3.1. *A **homotopy** is a map H that morphs f to g continously over a time interval $t \in [0, 1]$*

$$H : \mathbb{C} \times [0, 1] \to \mathbb{C}$$
$$f \longmapsto g$$

The homotopy itself is easy to build. We just consider $H(t) = (1-t)f + \gamma t g$. This says that at time $t = 0$ we have f and at time $t = 1$ we have g, almost. There are a couple of caveats here to discuss. One is that when we actually compute these homotopies, we start at time $t = 1$ and work backwards to time $t = 0$. In this way we are able to start from our simpler g and work back to the f that we want to solve. The second is the γ that mysteriously showed up in the homotopy. This is known as the γ trick [11]. We choose a random $\gamma \in \mathbb{C}$ and multiply this by our simple g. This allows for a probability one guarantee that our paths with not cross and, since it is a coefficient, it does not affect the roots of g.

Now we have a continuous morphing from g to f and for all solutions x_i of g there exists a path from $t = 1$ to $t = 0$ ending at either a solution of f or at ∞. The next step for us is to track each path such that we end at the solutions for which we are searching.

2.3.1 Predictor and Corrector

To track these paths, we call upon what are known as predictor/corrector methods. We start at $t = 1$ with Euler's method to take a step along the tangent line of the path. This moves us "forward" (we are actually moving backwards from 1 to 0), but the tangent direction might take us away from the path. So, we must correct back. Here we freeze t and employ Newton's method from calculus in order to bring us back to the path. We iterate this process of predicting and correcting until we have tracked the path back to the solution at $t = 0$. One question is, how far along the tangent should we step before we correct back? Another question is, with how much precision should we follow these paths? These questions relate directly to the *condition number* of the path at a particular point. This number

will give us an idea as to how our path is proceeding [11]. As t approaches zero, this also will provide a way to tell if a particular path is moving off to ∞.

Definition 2.3.2. *Consider F a system of polynomials and $\hat{x}$ a possible solution for F. We can consider the jacobian matrix, J, of our system F evaluated at our point $\hat{x}$, $J_{\hat{x}}(F)$. The* ***condition number*** *at $\hat{x}$ is the ratio of the largest singular value of the jacobian matrix to the smallest.*

With the condition number in hand, we begin to see how much precision we might need and how far we might be able to step. Such measures are already built into our favorite solver *Bertini* [2] in the form of adaptive multiprecision and adaptive step length. These, however, are technical considerations beyond the scope of this paper.

What this condition number does for us is to give a measure for how close we are to the nearest singular matrix. If the condition number is very large, this means that the value is very near a singular matrix and we need more precision. If the condition number is close to one, the value is not near a singular matrix and therefore easier to track. When solutions at ∞ are singular, they can cause many computational problems. One is that as we approach $t = 0$, we get condition numbers that are very high, thus more precision is needed. Using anything more that the lowest precision in a program like *Bertini* is very expensive. This ill-conditioning of singular infinite paths means that they cause a lack of efficiency. Even if infinite paths are not singular, they can cause problems. For instance, there is the matter of when to truncate such infinite paths. That is, how long should we track these paths before we decide that they are infinite. If we do not follow them long enough, we may end up truncating a solution that was not infinite. If

we follow for too long, we will end up wasting computing power and time on a lost pursuit.

2.3.2 Square Systems

One last consideration that has been thus far overlooked is that when using homotopy continuation methods, we assume the system is square. That is, the number of equations matches the number of variables. In general, polynomial systems are not square. In practice, polynomial systems are actually far from square. This is not a problem as we are able to square any system with a very simple method. Consider a system of N equations in k variables

$$F = \begin{cases} f_1(x_1, \ldots, x_k) \\ f_2(x_1, \ldots, x_k) \\ \vdots \\ f_N(x_1, \ldots, x_k) \end{cases}$$

with $N > k$. Then we wish to build a system $\hat{F}$ that is a $k \times k$ system with the same set of zeros as F. We accomplish this by taking linear combinations of the f_i's. This looks like

$$\hat{F} = \begin{cases} \hat{f}_1 = \alpha_{1,1} f_1 + \ldots + \alpha_{1,N} f_N \\ \hat{f}_2 = \alpha_{2,1} f_1 + \ldots + \alpha_{2,N} f_N \\ \vdots \\ \hat{f}_k = \alpha_{k,1} f_1 + \ldots + \alpha_{k,N} f_N \end{cases}$$

where the $\alpha_{i,j}$ are random coefficients in $\mathbb{C}$. How can we guarantee that we still get the same zeros? There is one theorem of Bertini that allows this squaring. For simplicity we will keep the current notation.

Theorem 2.3.1. *(Bertini) Solutions of F are still solutions of $\hat{F}$.*

Proof:

If a value $\bar{x}$ is a solution to F, then $f_1(\bar{x}) = f_2(\bar{x}) = \ldots = f_N(\bar{x}) = 0$. Since each $\hat{f}_i$ is a linear combination of the f_j's, we have that $\hat{f}_1(\bar{x}) = \hat{f}_2(\bar{x}) = \ldots = \hat{f}_k(\bar{x}) = 0$. Thus each solution of F is a solution of $\hat{F}$.

□

NOTE: Bertini's theorem actually says more than this. This piece is all that is necessary for our purposes.

On the other hand, we may end up with solutions to $\hat{F}$ that are not solutions to F. These solutions may lead off to ∞ or they may lead to what are called Bertini junk points. The infinite paths, as we have shown, are difficult to track. The Bertini junk points are computationally much less expensive. They are not infinite and it is an easy test to see that they are not solutions to F and discard them. It will be our goal in preconditioning to remove the infinite paths, possibly at the expense of creating more paths that lead to Bertini junk points. Most importantly, any polynomial system with the number of equations greater than the number of variables can be made square. Now, we have the tools to use homotopy continuation methods.

Chapter 3

DUAL SPACES AND H-BASES

In this chapter we will look at Dual Spaces and H-bases as a means to precondition polynomial systems. These ideas will allow us to avoid the computational problems that arise from the use of Gröbner Bases. Gröbner basis methods allow for many computations symbolically but fail to stay stable when working with numerical approximations. We will be able to use a similar idea in the form of H-bases without the extra criteria that cause these problems. Specifically, H-bases will not require a division algorithm involving the leading terms and leading coefficients. This means that by working with numerical linear algebra, we can even work with numerical approximations.

We will use the dual space of an ideal in order to find the Hilbert function of the system. This will determine a first stopping criterion for our algorithm as it will identify an H-basis. We do this by first homogenizing the ideal generated by our system. At each step of this process, we are able to read off the Hilbert function in the different degrees. When this stabilizes, we are able to move forward to the next step. The H-basis will then be a new homogeneous system of polynomial equations that admits the same affine variety. We can iterate this process to create each new H-basis until we get the Hilbert function to stabilize to the Hilbert polynomial, this will be our second stopping criterion. The H-basis represented at this step will provide our preconditioned system. This new system will allow us to compute the solutions to the system more efficiently by removing paths that go off to ∞.

At most we may get extra Bertini junk points. Otherwise this method has the potential to create a preconditioned system that admits the exact number of paths. We will start with some basic definitions from linear algebra in order to familiarize ourselves with the notation.

3.1 Linear Algebra

The fundamental constructs that we will use for dual spaces and H-bases are rooted in linear algebra. There are different ways of defining dual spaces. We can follow [12] and define a dual space in terms of vector spaces. First, recall our notation from Chapter 2. Let $\{f_1, \ldots, f_s\}$ be a system of polynomial equations with $f_i \in \mathbb{C}[x_1, \ldots, x_n]$. Again, we will consider $I \subset \mathbb{C}[x_1, \ldots, x_n]$ the ideal generated by our system, $I = < f_1, \ldots, f_s >$.

Definition 3.1.1. *The* ***dual space*** V^* *of a vector space* V *is the vector space of all linear functionals* $\quad L : V \to \mathbb{C}$

This definition is very general and difficult to use for computations. We want to find an equivalent definition that encodes all of this information in a more usable form. We can encode the linear functionals into our computations by using the Macaulay matrix. In order to do this, we must define the differential operators ∂_α. For $\alpha \in \mathbb{Z}^n_{\geq 0}$ we have $|\alpha| = \alpha_1 + \ldots + \alpha_n$ and $\alpha! = \alpha_1!\alpha_2! \cdots \alpha_n!$ Now we say the differential operator is

$$\partial_\alpha = \frac{1}{\alpha!} \frac{\partial^{|\alpha|}}{\partial x^\alpha}$$

For $g \in \mathbb{C}[x_1, \ldots, x_n]$ and $y \in \mathbb{C}^n$ we have $\partial_\alpha[y](g) = (\partial_\alpha g)[y]$. This says that the differential operator evaluated at our y value and then applied to our polynomial

g is the same as applying the operator to g and then evaluating at y, [7]. Now we can define our Macaulay matrix using this notation.

Definition 3.1.2. *The* ***Macaulay matrix*** $M_d(I)$ *of degree* d *for an ideal* I *is the matrix whose row entries are the functions* $f_1 \dots f_s$ *of* I *and whose column operators are the differential operators* ∂_α *of degree* d*, i.e.* $|\alpha| = d$*.*
In other words $m_{i,j} = \partial_{\alpha_j}(f_i)$*.*

This definition isn't exactly precise. If the functions $f_1, \dots, f_s$ are not all of degree d we must first raise them to degree d. This entails multiplying the f_i's by all monomials of degree equal to the difference of d and the degree of f_i. In the end, we may end up with many more functions in our rows.

Another definition of the degree d dual space of I is the space of differential operators of degree d that vanish on I. If we have the space of differential operators on I encoded into our Macaulay matrix, we need to look at the vectors that vanish on I.

Definition 3.1.3. *The* ***null space*** $N(A)$*, of a matrix* A*, is the set of vectors* $v \in \mathbb{C}^n$ *such that* $A \cdot v = 0$*.*

We are now able to use these ideas from linear algebra to give a better defintion of the dual space of degree d of an ideal I. When we find the nullity of a matrix A, we can find a basis for $N(A)$. This will provide for us a $\mathbb{C}$-basis of the dual space called the *dual basis*.

Definition 3.1.4. *The* ***(Macaulay) dual space*** *of an ideal* I *is the null space of the Macaulay matrix* $N(M_d(I))$*.*

NOTE: This is not the most general definition for either the Macaulay matrix or the dual space. This is, however, the most useful version for our algorithm. From this dual space definition we can build a dual basis for the quotient ring $\mathbb{C}[x_0, \ldots, x_n]/I$ where I is the ideal generated by our system of equations.

3.2 Hilbert Function

We will now to relate this to the Hilbert function in order to give a stopping criterion for finding a correct $\mathbb{C}$-basis for our dual space. Following [3] we can define the Hilbert function for a homogeneous system of equations. Since we are going to homogenize our system, we can consider the homogeneous ideal $I^h \subset \mathbb{C}[x_0, \ldots, x_n]$ in $n+1$ variables.

Definition 3.2.1. *If K is a finitely generated graded module over $R = \mathbb{C}[x_0, \ldots, x_n]$ then the **Hilbert function** $H_K(d)$ of degree d is defined to be the dimension of the degree d homogeneous part K_d of K over $\mathbb{C}$, $\quad H_K(d) = dim_{\mathbb{C}} K_d$.*

This definition gives us a way now to relate the Macaulay dual space defined in the last section to our ideal. Since this is a graded module over our ring, R, we can consider the homogeneous part in degree d as a vector space over $\mathbb{C}$, [3]. In this way, we remember that we are working over graded modules but we consider our spaces as vector spaces over a field. This allows us to use linear algebra and numerical linear algebra.

Proposition 3.2.1. *The Hilbert function in degree d of I^h, is equal to the dimension of the null space of the Macaulay matrix of I^h in degree d.*

$$H_{I^h}(d) = dim_{\mathbb{C}} N(M_d I^h)$$

Proof:

By definition of the Hilbert function in degree d of the homogeneous ideal I^h, $H_{I^h}(d) = dim_{\mathbb{C}}(\mathbb{C}[x_0, \ldots, x_n]_d) - dim_{\mathbb{C}}(I^h_d)$. *From linear algebra we know that the dimension of a dual vector space is equivalent to the dimension of the original vector space as long as the vector space is finite-dimensional. The dimension in* $\mathbb{C}$ *of* I^h *is the rank of the Macaulay matrix. By the rank-nullity theorem, we have that the difference between the dimension of the whole space and the rank is the null space of the Macaulay matrix,* $N(M_d I^h)$.

□

3.3 H-bases

H-bases, first defined by Macaulay, give us a way to find usable bases for our computations without the extra conditions of Gröbner bases. The primary difference lies in the fact that H-bases do not require a division using leading terms and leading coefficients. Instead, H-bases are only ordered by degree. Macaulay called them H-bases for homogeneous [8, 9]. Once we have an H-basis, it will be a homogeneous system that will correspond to the Hilbert function of our original system. If we iterate this process until a stopping criterion, we create an H-basis that corresponds to the Hilbert polynomial. Our first simple example of an H-basis will just be the homogenized system, if it is not already homogeneous. We will use the notation H_0 for this H-basis. Each succesive H-basis will use similar notation. First, we will need a definition of an H-Basis. We will follow the definition and notation from [8].

Definition 3.3.1. *Consider a system of polynomials* $F = \{f_1, \ldots, f_s\}$ *with* $f_i \in \mathbb{C}[x_1, \ldots, x_n]$. *Let* $G = \{g_1, \ldots, g_r\}$ *be a finite set of polynomials with* $g_i \in \mathbb{C}[x_1, \ldots, x_n] \setminus \{0\}$. *We say* G *is an* ***H-Basis*** *for* $I = < f_1, \ldots, f_s >$ *if for all* $p \in I$, *there exists* $h_1, \ldots, h_r$ *such that*

$$p = \sum_{i=1}^{r} h_i g_i \quad \textbf{\textit{and}} \quad (deg(h_i) + deg(g_i)) \leq deg(p) \quad \text{for } i = 1, \ldots, r$$

Here we see that the criteria for an H-basis depends solely on the degree of the polynomials. We are going to compute H-bases in a step by step process. We start with H_0 which is now our homogenized system. In order to get to H_1 we consider the Macauley matrix of our homogenized system. One question for our calculation is, in what degree should we compute the Macaulay matrix? At each step, there is a lower bound on this number.

The smallest degree in which we can compute the Macaulay matrix is the largest degree of the f_i's. Consider $d = \max\{deg(f) \mid f \in f_1, \ldots, f_s\}$. We will then start in degree d and compute the null space of the Macaulay matrix. We are looking for the dimension of the null space to stabilize. Using this construction, we are looking for the dimension of the null space to become constant. For different constructions, we would look for the difference in the dimensions of the null space to become constant. We compute the null space of the Macaulay matrix in degree $d+1, d+2, \ldots$ until we reach degree k such that $dim_{\mathbb{C}} N(M_k I^h) = dim_{\mathbb{C}} N(M_{k+1} I^h)$. At this point, we can work in degree k. Occasionally, this method will yield an H-basis that does not work out in the end and we must instead use degree $k+1$. For a possible heuristic, it might be best to use $k+1$ in every case, though this may not give optimal performance. This phenomena will be demonstrated in the chapter of examples and discussed in the conclusion.

From this null space computation, we obtain a dual basis in degree k. It is here that we want to quotient out our ideal by the homogenizing variable x_0. We accomplish this by trimming our basis vectors to remove any monomials that do not include x_0. In the setup of the algorithm, this is accomplished by ordering the monomials with a lexicographic ordering with $x_0 > x_1 > \ldots > x_n$. Once we have trimmed the vectors, we can put the new trimmed vectors into a new matrix $\hat{M}$ and take the null space of the transpose, $N(\hat{M}^T)$. What remains is a new basis for our ideal I^h. This new basis is in degree $k-1$. This will be our H_1, the next H-basis. Taking the transpose of $\hat{M}$ puts the equations into vectors and the null space calculation removed one level of x_0 components. This also has the effect of moving back to our original space as the dual of the dual space V^* is the original space, $(V^*)^* = V$.

Now that we have H_1, the lowest degree that we may compute the Macaulay matrix in is $k-1$. If, indeed, degree $k-1$ again yields an H-basis, we have cut the degree of our system down again and our new system is in degree $k-2$. This corresponds to removing paths when using a total degree start system for homotopy continuation. Now that we have our method for finding successive H-bases, the next step is to find where to stop. At what point do we have a fully saturated system?

Since we are working with zero dimensional systems, it is known that the Hilbert polynomial is constant. Our second criterion is for the degree of the H-bases to stabilize to a number. Once we have reached this point, we have fully saturated with respect to the homogenizing variable. This means that we have reached the Hilbert polynomial and the degree of the null space will be the exact number of solutions. The final H-basis will represent our preconditioned system.

The beauty of this is that we now have the exact number of solutions and a system that is preconditioned. When solving this system with homotopy continuation, we have fully removed all infinite paths.

3.4 Computational Method

ALGORITHM:

INPUT: A system of polynomials, $F = \{f_1, \ldots, f_s\} \subset \mathbb{C}[x_1, \ldots, x_n]$
OUTPUT: A preconditioned homogeneous system $\{h_1, \ldots, h_r\} \subset \mathbb{C}[x_0, \ldots, x_n]$

- **Step 0:** Homogenize system, $F \mapsto F^h$, this is H_0
- **Step 1:** Let $J = H_0$
- **Loop 1:** For i from 0 to ℓ such that $dim(H_\ell) = dim(H_{\ell+1})$ do
  - **Loop 2:** For d from lowest degree (from above) to k such that $dimN(M_k(J)) = dimN(M_{k+1}(J))$
    - **Step:** Build Macaulay matrix $M_d(J)$ in degree d
    - **Step:** Find a basis $\hat{N}$ for the null space $N(M_d(J))$
    - **Step:** If dimension stabilizes, break out of Loop 2, else $d \to d+1$
    - End for
  - **Step:** Trim basis vectors and find the null space of $\hat{M}^T$, ($\hat{M}$ is trimmed matrix)
  - **Step:** Convert to polynomials , this is H_i, and set $J = H_i$
  - End for

Chapter 4

H-BASIS EXAMPLES

Through dual space computations, we can now produce H-bases. These H-bases give defining equations for the solution set that cut away extraneous paths to ∞. This chapter gives some examples of H-basis computations and the null space criteria provided by the Hilbert function and the Hilbert polynomial.

4.1 Preliminary Examples

We begin this section with some examples which each have only one solution [1]. We present a comparison to the most general root counts for systems as described in chapter 1. This includes the total degree bound and the mixed volume calculation where available.

4.1.1 Example 1

We consider the system:

$$f = \begin{cases} x^2 - 9 \\ \\ xy + 3y - 1 \end{cases}$$

By the total degree count, we have a possible 4 roots. The mixed volume calcuation tells us that there are possibly two roots. With the H-basis calculation, we can show that there is one unique root. We can also find the defining equations for

said unique root. Table 4.1 shows the progression of the H-basis computations in different degrees. As the degree of the null space stabilizes, we move forward to the next H-basis. The dimension of the null space that gives the correct value is shown in blue. Sometimes a nullity stabilizes at a lower degree but does not give the correct H-basis. This nullity is here denoted in red. A clear example of this is shown in the table as H_3', which does not work out correctly. This step came from choosing the equations that came from degree 2 of H_2 rather than degree 3. If we instead choose degree 3, then the set labeled H_3 gives a correct H-basis. Here this misstep comes from the fact that the number of equations is not enough to have unique solutions. This phenomenon of the red numbers will be discussed in the final chapter and is the subject of future work.

H-basis	degree 2	degree 3	degree 4
H_1	null = 4 0 equations	null = 4 3 equations	null = 4 7 equations
H_2	null = 3 1 equation	null = 3 4 equations	null = 3 8 equations
H_3'	null = 3 1 equation	null = 4 3 equations	null = 5 6 equations
H_3	null = 2 2 equations	null = 2 5 equations	null = 2 9 equations
H_4	null = 1 2 equations	we get a unique solution from this set of equations	

Table 4.1: H-Basis steps

4.1.2 Example 2

Our next example is a little more complicated. We consider the system:

$$g = \begin{cases} x^2 + 6xy + 4xz + 9y^2 + 12yz + 4z^2 - 9 \\ 4x^2 + 10xy + 9xz - 6y^2 - yz + 2z^2 - 1 \\ 5x^2 + 12xy + 9xz - 15x - 9y^2 - 9yz + 9y - 2z^2 + 3z - 1 \end{cases}$$

Here both the total degree and the mixed volume calculations show a possible eight roots. Due to the large number of differing monomials, the mixed volume calculation does not cut down the number of possible roots. We will precondition the system to produce only one path. This can be shown as we reach the Hilbert polynomial. When we get to our last H-basis, as shown in Table 4.2, the dimension of the null space is one. This removes the seven extraneous paths that go off to ∞.

H-basis	degree 2	degree 3	degree 4
H_1	null = 7 0 equations	null = 8 6 equations	null = 9 16 equations
H_2	null = 4 2 equations	null = 4 8 equations	null = 4 18 equations
H_3'	null = 3 2 equations	null = 4 7 equations	null = 5 16 equations
H_3	null = 2 3 equations	null = 2 9 equations	null = 2 19 equations
H_4	null = 1 3 equations	we get a unique solution from this set of equations	

Table 4.2: H-Basis steps

Here we have the correct null space dimension criterion shown in blue. If we again take a misstep at H_2 we end up with an incorrect H-basis. This is denoted H_3' and the wrong choice of nullity is again in red.

4.2 Reimer 3

We consider the system

$$h = \begin{cases} -1 + 2x^2 - 2y^2 + 2z^2 \\ -1 + 2x^3 - 2y^3 + 2z^3 \\ -1 + 2x^4 - 2y^4 + 2z^4 \end{cases}$$

This system arises from a problem posed by Reimer [10]. Using a total degree bound, the original system has 24 paths with 12 of them leading off to ∞.

H-basis	degree 3	degree 4	degree 5	degree 6	degree 7
H_1	N/A	null = 20 5 eqs	null = 23 16 eqs	null = 24 36 eqs	null = 24 64 eqs
H_2	N/A	N/A	null = 20 19 eqs	null = 20 40 eqs	null = 20 68 eqs
H_3	N/A	null = 16 7 eqs	null = 16 21 eqs	null = 16 42 eqs	
H_4	null = 13 2 eqs	null = 14 9 eqs	null = 14 23 eqs		
H_5	null = 11 2 eqs	null = 12 9 eqs	null = 12 23 eqs		
H_6	null = 11 2 eqs	null = 12 9 eqs			

Table 4.3: Reimer 3

As shown in Table 4.3, the dimension of the null space of the Macaulay matrix is 24 when we start into H_1. As the nullity stabilizes and we step through the bases, we get the dimension of the null space down to 12 exactly. This H-basis then produces 27 paths with 15 Bertini junk points and zero infinite paths. Through the H-basis computations we have been able to remove all of the infinite paths and we only added an additional three paths.

4.3 A Problem From Economics

Here we look at a problem arising from economics. We consider the system:

$$f = \begin{cases} x_1x_5 + x_1x_2x_5 + x_2x_3x_5 + x_3x_4x_5 - 1 \\ x_2x_5 + x_1x_3x_5 + x_2x_4x_5 - 2 \\ x_3x_5 + x_1x_4x_5 - 3 \\ x_4x_5 - 4 \\ x_1 + x_2 + x_3 + x_4 + 1 \end{cases}$$

This system has a total degree count of 54 possible solutions. Of these 54 paths, only eight actually lead to solutions. The extra 46 paths are all going off to ∞ when tracking using homotopy continuation. It is our goal to remove the 46 paths to ∞. The table shows that we are able to get the rank of the null space lowered. This, corresponding to the Hilbert function in each degree, gives us the bound on the number of possible solutions at each step. In the last step in the table, the dimension of the null space is lowered to eight which is the exact number of solutions. Our second criterion says that since the dimension of the null space

stabilizes here, we have reached saturation. This H-basis has 32 paths and the 24 extraneous paths are not infinite but instead are Bertini junk points.

H-basis	degree 2	degree 3	degree 4	degree 5
H_1	N/A	null = 27 8 eqs	null = 40 35 eqs	null = 52 99 eqs
H_2	N/A	null = 21 12 eqs	null = 27 46 eqs	null = 33 116 eqs
H_3	null = 9 1 eq	null = 10 12 eqs	null = 10 47 eqs	
H_4'	null = 9 1 eq	null = 10 12 eqs	null = 10 47 eqs	
H_4	N/A	null = 9 13 eqs	null = 9 48 eqs	
H_5	null = 8 1 eq	null = 8 13 eqs	null = 8 48 eqs	

Table 4.4: H-Basis steps

It is worth mentioning that the mixed volume calculation for this system gives the exact number of roots at eight. But, the time spent calculating the mixed volume may outweigh the benefits of tracking only eight paths.

This particular example is a curiosity in that the dimension of the null space in H_1 doesn't stabilize either to a single number or to a linear equation. The difference between the dimension in null spaces seems to be quadratic. Degree four was chosen because it represented a minimum for the parabola defined by the quadratic change. The fact that the nullity didn't stabilize is another future consideration.

Chapter 5

CONCLUSION

While the ideas for dual spaces and H-bases have been around for quite a while, only recently has there been any progress in using them to effectively compute operations on ideals. With the computing power now available, and this method's natural parallel extension, these ideas are quickly becoming a viable tool for such operations. Our goal is to use these methods to precondition our polynomial systems for use in homotopy continuation methods. There are several methods for reducing the number of paths and the time per path. Most of these are concerned with building a good enough start system. This method works but may end up being more computationally expensive if the start system is too difficult to solve easily. What we have done is to cut down the original system so that we may use the simplest start system, total degree, and still track fewer paths with none of the paths being infinite. In the end this may produce more paths and we must test for feasibility. There are a few questions still to be answered. How many H-basis calculations should we do before the costs outweigh the benefit? How can we streamline the calculations for maximum efficiency? Why does our first stopping criterion occasionally give a false H-basis? How could we deal with positive dimensional varieties?

5.1 Computation Time

The algorithm presented is not currently designed to be the most efficient possible. This algorithm, instead, shows that we can completely cut out all of the infinite paths for homotopy continuation. The method has the added bonus that it can, under the right circumstances, cut down the total number of paths to be tracked. In this way, we are tracking fewer paths and each path is less computationally expensive. For medium to large problems, this method may be particularly valuable. For small problems, it may be better not to precondition at all as the difference in computation time is rarely better than seconds saved and at times not at all.

A heuristic is being worked on as to when we should stop H-basis calculations. If it is more cost effective, we may choose to stop the H-basis computations before saturation. This means that there will be some paths left that are infinite. These are represented as solutions where our homogenizing variable is zero, $x_0 = 0$. These can then be filtered out as non solutions. The cost-benefit ratio is not quite clear yet but a useable criterion is our next step.

Another improvement to the algorithm comes in the form of reducing the sizes of the Macaulay matrices. Zeng [13] provides this with the *Closedness Subspace Method*. This method will allow us to define the Macaulay matrix using the exact differential operators that are needed for each computation. Currently, we are using a more general method that includes all differential operators of a particular degree. It may be that some of these differential operators have no effect on the calculation. Thus, the size of the Macaulay matrix can be reduced and the efficiency of the null space computations increased.

5.2 Discussion

One obstacle in this algorithm is when we reach a false H-basis. This was shown in several examples in Chapter 4. One explanation for this red number phenomena was that the number of equations and the number of variables made a single solution impossible. This certainly happened in 4.1.1 and 4.1.2. Clearly two equations in four variables was not enough to carve out a single solution. The misstep in 4.3, however, does not share this same characteristic. At the misstep, the dimension of the null space stabilizes and following the algorithm says that we should consider the 12 equations in degree three. Doing so leads to false solutions. One possible explanation is that the trimming operation removes too much information when the degree is minimal. It is an easy fix to just use the next H-basis. The question remains: How do we detect such missteps? This is still an open question.

Another obstacle comes in the form of an exact first stopping criterion. In most examples, we know to stop because the dimension of the null space stabilizes to a single number. In 4.1.2, the dimension of the null space does not stabilize to a number, but instead to a linear function. This criterion is one for a slightly different version of this algorithm but worked in this case. Again in example 4.3, the dimension of the null space appears to be a quadratic function. Choosing the minimum value gave a correct H-basis. Yet, there was no specific criterion for doing this. At a later step in the algorithm, we are again faced with a linear function rather than a constant. It seems as though the function is dropping by degrees from quadratic to linear to constant. More research will need to be dedicated to discovering the cause of this phenomenon.

5.3 Future Work

There are a number of places to take this research. Future work will mostly be concerned with creating better efficiency for the algorithm. This will include implementing the closedness subspace method and finding a reasonable heuristic for stopping the H-basis computations. Another possible method for speed up would be to use only the monomials involving the homogenizing variable to raise the degree for the Macaulay matrix. In preliminary tests, this has shown valuable speed up but further tests are necessary before this can be shown as viable.

Other research will include the case for positive dimensional varieties. Here there is much more work to be done in order to find a reasonable stopping criterion. Preliminary results from the twisted cubic give hope but not yet a definitive method. There is also the concern for when the dimension of the null space does not behave as expected. We hope to broaden our criterion for stabilization or find the root of this occurence. Only further work will tell.

Acknowledgements

Thanks first and foremost to my advisor, Dan Bates. Further thanks go to Jon Hauenstein for his guidance and to my wife Heather for her support. Also, this work wouldn't be possible without the support of the CSU Math Department.

Bibliography

[1] AMS-SIAM, *Computing hilbert functions using dual spaces*, Jan 2010.

[2] D. Bates, A.J. Sommese, C.W. Wampler, and J. Hauenstein, *Bertini, software for numerical algebraic geometry*, Available at http://www.nd.edu/ sommese/bertini/.

[3] D. Cox, J. Little, and D. O'Shea, *Using algebraic geometry*, Springer, 1998.

[4] ———, *Ideals, varieties, and algorithms*, third ed., Springer, 2007.

[5] D.S. Dummit and R.M. Foote, *Abstract algebra*, third ed., John Wiley & Sons, Inc., 2004.

[6] D.R. Grayson and M.E. Stillman, *Macaulay2, a software system for research in algebraic geometry*, Available at http://www.math.uiuc.edu/Macaulay2/.

[7] J. Hauenstein, *Algebraic computations using macaulay dual space*, 2010, preprint.

[8] H.M. Möller and J. Sauer, *H-bases for polynomial interpolation and system solving*, Advances in Computational Mathematics **12** (2000).

[9] H.M. Möller and T. Sauer, *H-bases 1: The foundation*, Vanderbilt University Press, 1999.

[10] M. Reimer, *Constructive theory of multivariate functions*, (1990).

[11] A.J. Sommese and C.W. Wampler, *The numerical solution of systems of polynomials arising in engineering and science*, World Scientific, 2005.

[12] H.J. Stetter, *Numerical polynomial algebra*, SIAM, 2004.

[13] Z. Zeng, *The closedness subspace method for computing the multiplicity structure of a polynomial system*, 2000.